validate Documentation

Release 0.1

David W. Fallis

Apr 23, 2020

Contents

CHAPTER 1

Contents

1.1 Quickstart

1.1.1 Installing

You can install validate with pip from pypi:

```
pip install validate
```

or the latest version directly from github

```
pip install git+https://github.com/swartn/validate.git
```

validate has primarily been developed and tested within the anaconda python distribution on Linux x86/x64 and Mac OSX. Windows is not supported.

You can (should) do this inside a virtual environment. In that case it will work without root privileges. If you are using anaconda see http://conda.pydata.org/docs/faq.html#env.

Dependencies

The external package Climate Data Operators (cdo) v1.6 or later is required. Python dependencies are in theory handled automatically by pip, but often things go smoother if you have these items available:

- Python 2.7x
- Climate Data Operators (cdo) v1.6 or later.
- netCDF4
- numpy 1.2.1 or later
- matplotlib
- pyyaml
- brewer2mpl

- ‘cmipdata <https://cmipdata.readthedocs.org/en/latest/>’_

validate has primarily been developed and tested within the anaconda python distribution on Linux x86/x64 and Mac OSX, where dependencies were installed with conda. Windows is not supported.

1.1.2 Using validate

After a succesful installation, you should move to a working directory (preferably empty). Be careful working in populated directories as validate may remove or modify files or directories with particular names.

To setup the configuration file use the command:

```
validate-configure
```

which will provide a conf.yaml file. Follow the examples in this file to specify the plots you desire, and make sure to modify the “data_root” variable to point to the data on your system. Once the changes have been made, save the conf.yaml file in your working directory and once use the command:

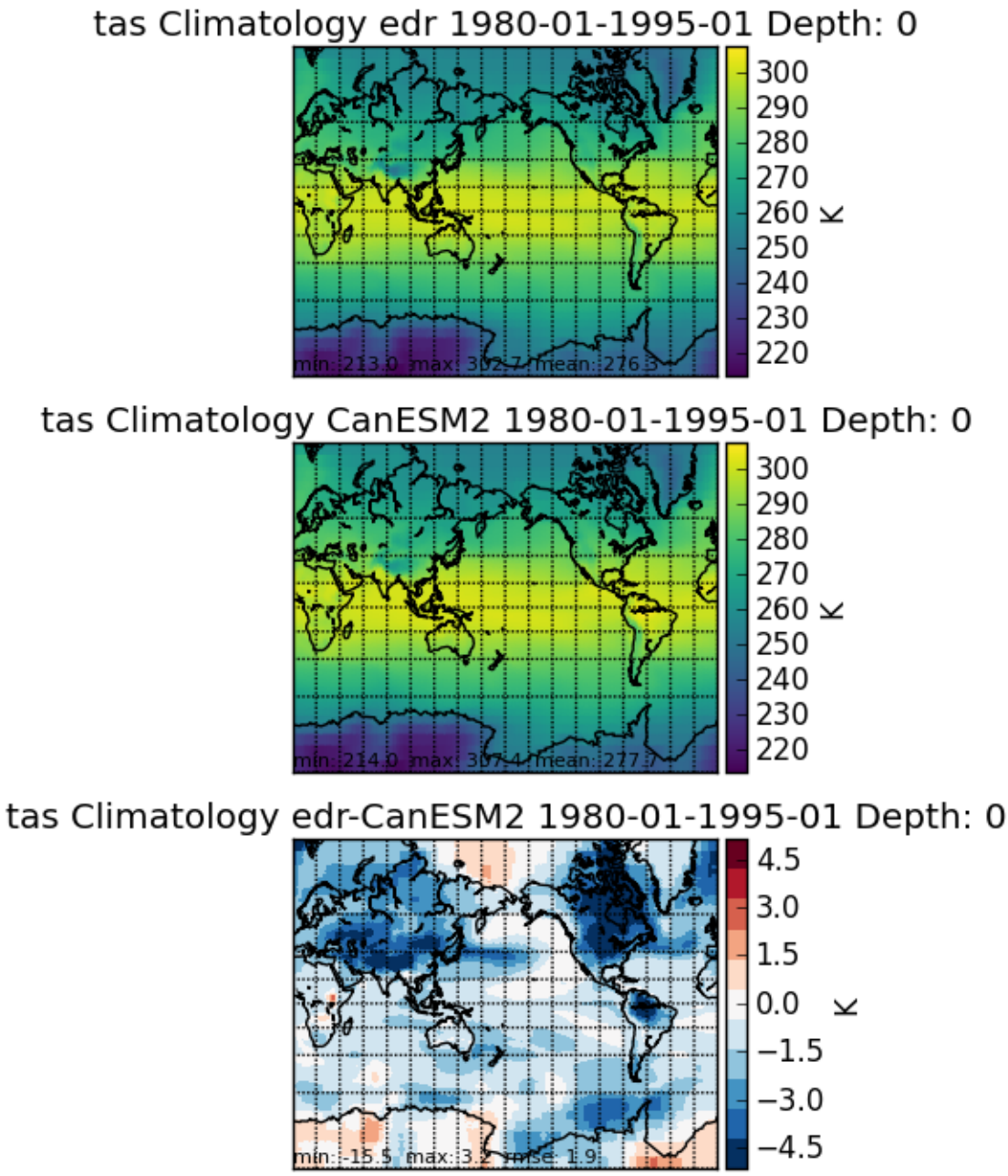
```
validate-execute -r [runID]
```

The generated plots can be found in the /plots directory and details about the plots in the /log directory.

1.2 Examples

1.2.1 Mercator Projection

For this example we want to make a color map comparison between the edr run ID and CanESM2 for the atmospheric surface temperature climatology between 1980 and 1995.



First use the command:

```
validate-configure
```

Then edit the conf.yaml file to the following:

```
run: 'edr'
experiment: 'historical'

defaults:
            dates:
              start_date: '1980-01'
              end_date: '1995-01'
```

(continues on next page)

(continued from previous page)

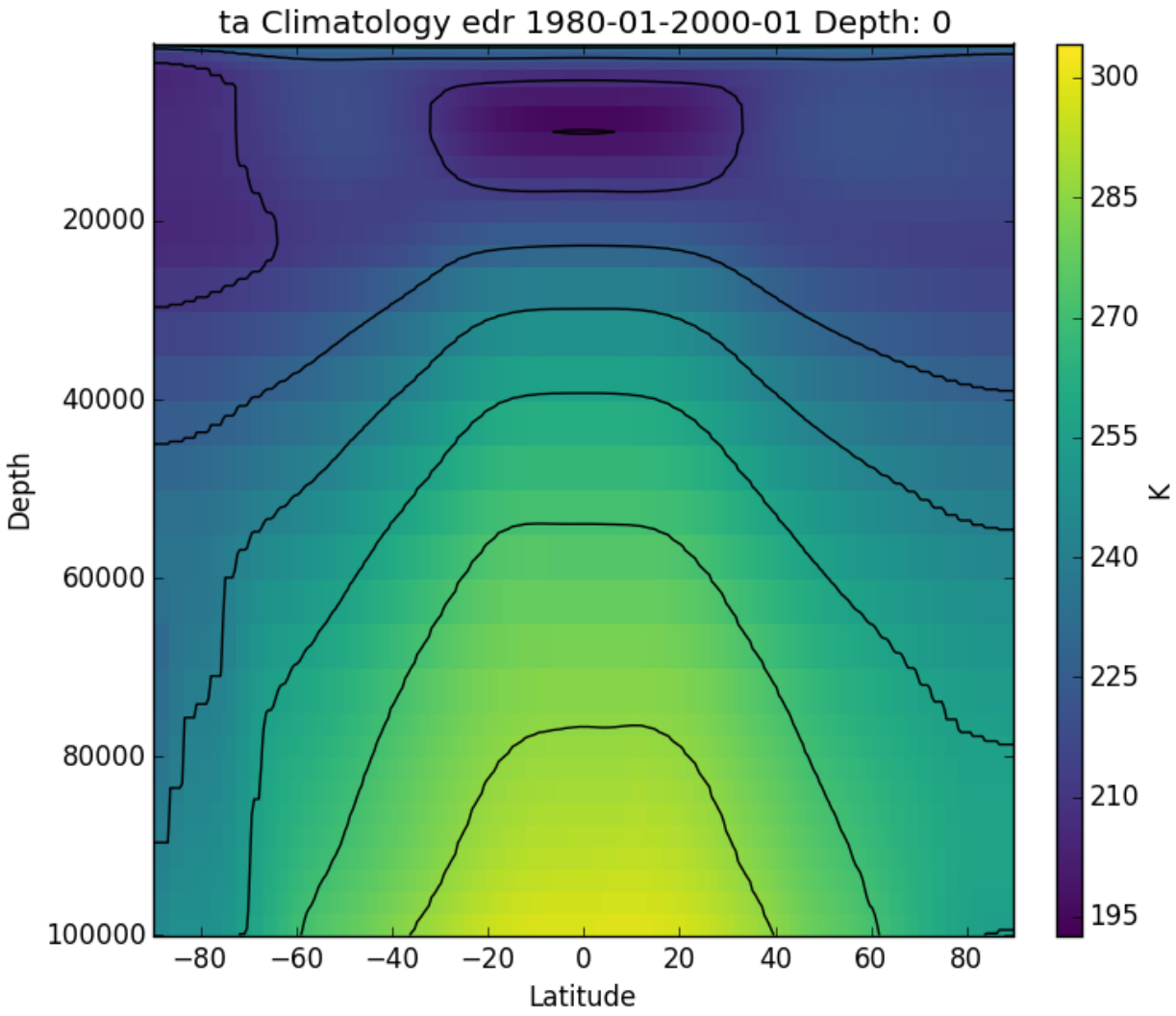
```
            png: True

plots:
        - variable: 'tas'
          plot_projection: 'mercator'
          data_type: 'climatology'
          comp_models:
            - CanESM2

delete:
          del_netcdf: False
          del_mask: True
          del_ncstore: True
          del_cmipfiles: False

direct_data_root: '/raid/rc40/data/ncs/historical-edr'
observations_root: '/raid/rc40/data/ncs/obs4comp'
cmip5_root: '/raid/ra40/CMIP5_OTHER_DOWNLOADS/'
```

Save the file and then use the command:

```
validate-configure
```

1.2.2 Zonal Mean Plot

For this example we want to make a color map of a zonal mean cross section of the atmospheric temperature of the climatology from 1980 to 2000 for the edr run ID.



First use the command:

```
validate-configure
```

Then edit the conf.yaml file to the following:

```
run: 'edr'
experiment: 'historical'

defaults:
            dates:
              start_date: '1980-01'
              end_date: '2000-01'
            png: True

plots:
        - variable: 'tas'
          plot_projection: 'zonal_mean'

delete:
```

(continues on next page)

(continued from previous page)

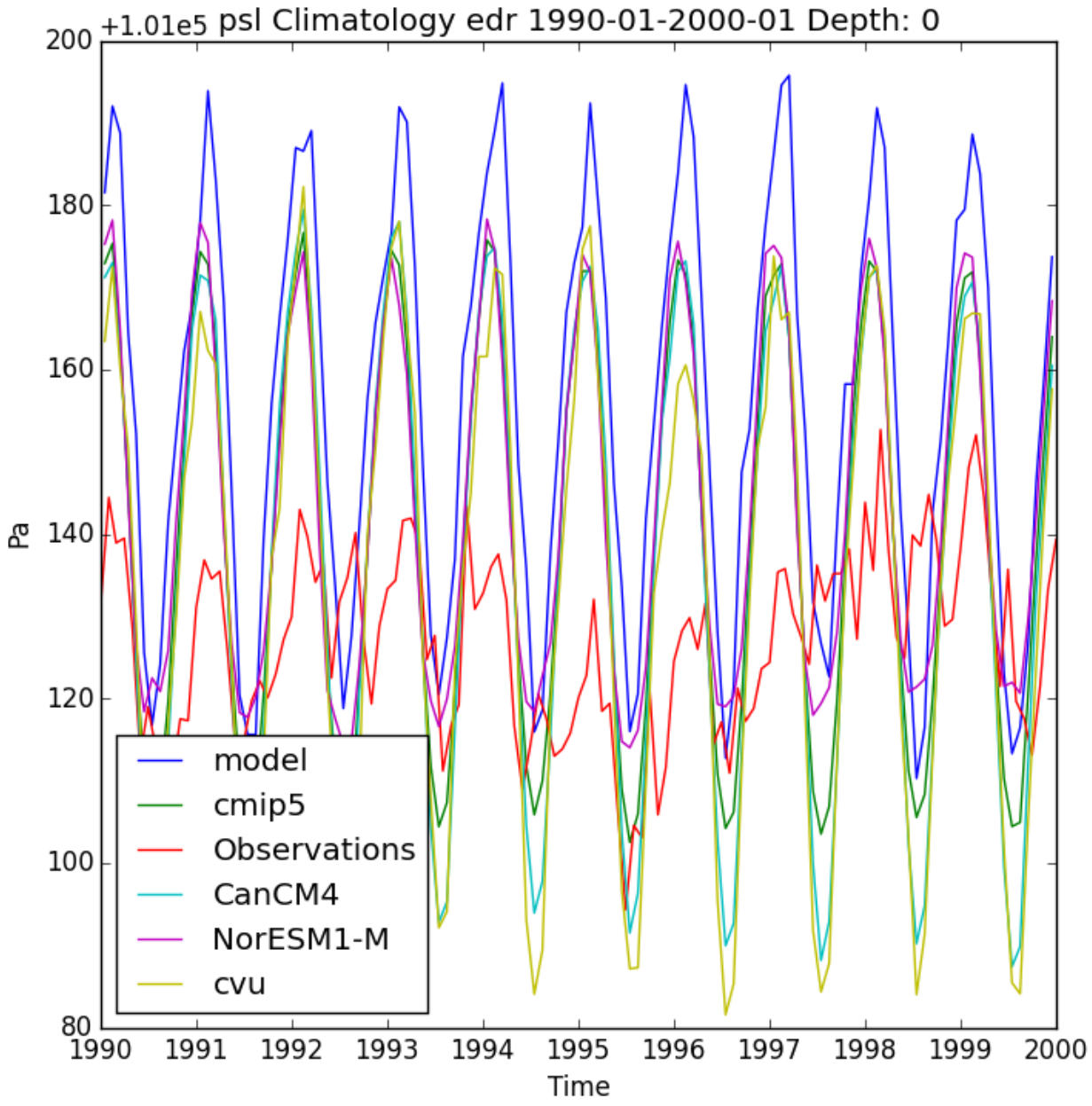
```
          del_netcdf: False
          del_mask: True
          del_ncstore: True
          del_cmipfiles: False

direct_data_root: '/raid/rc40/data/ncs/historical-edr/'
observations_root: '/raid/rc40/data/ncs/obs4comp'
cmip5_root: '/raid/ra40/CMIP5_OTHER_DOWNLOADS/'
```

Save the file and then use the command:

```
validate-configure
```

1.2.3 Time Series Plot

For this example we want to make a time series plot of the atmospheric pressure at the surface for the climatology from 1980 to 2000.



First use the command:

```
validate-configure
```

Then edit the conf.yaml file to the following:

```
run: 'edr'
experiment: 'historical'

defaults:
            dates:
              start_date: '1990-01'
              end_date: '2000-01'
            png: True
```

(continues on next page)

(continued from previous page)

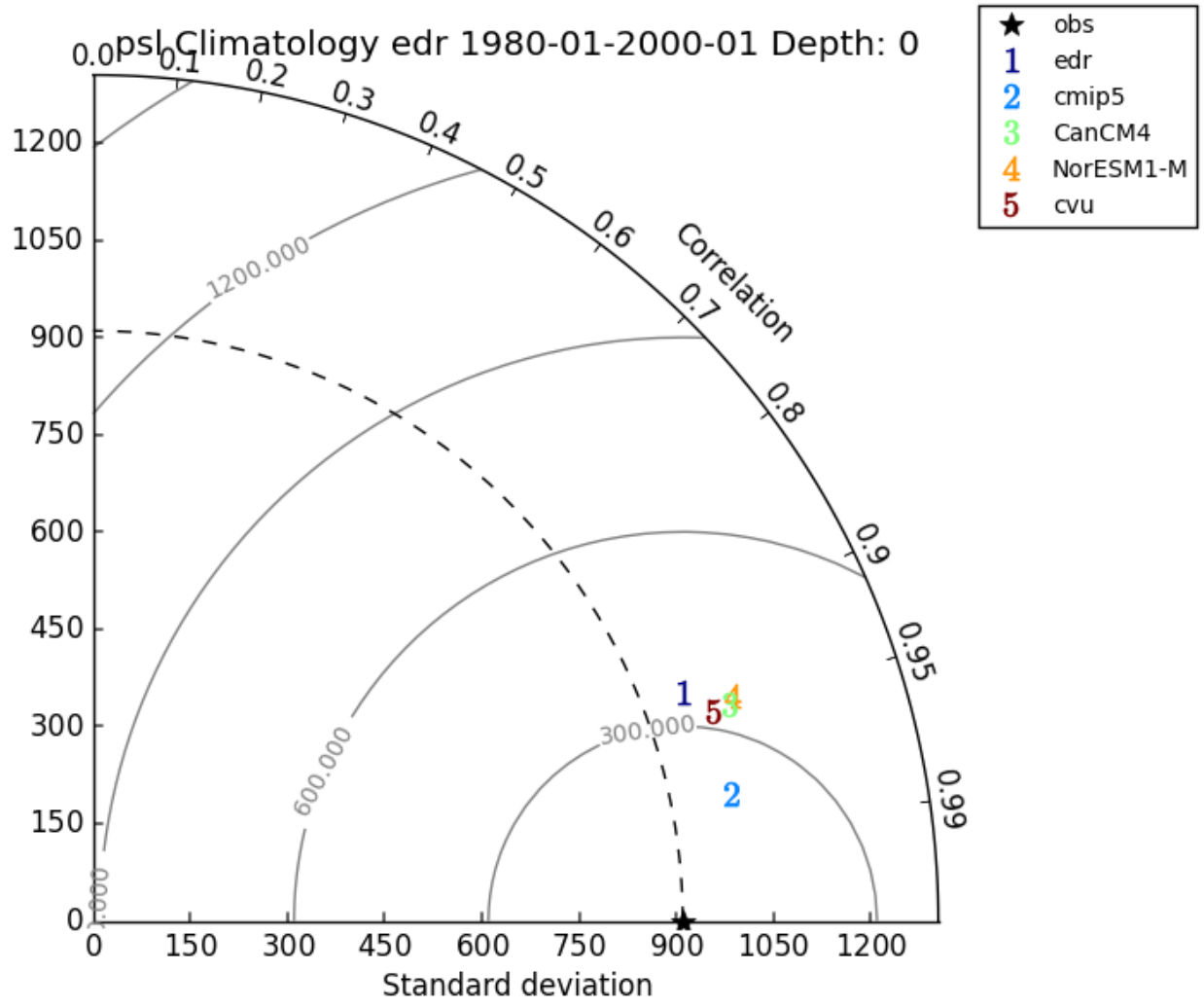
```
plots:
        - variable: 'psl'
          plot_projection: 'time_series'
          comp_obs:
            - 20CR
          comp_cmips: 'all'
          comp_ids:
            - cvu

delete:
          del_netcdf: False
          del_mask: True
          del_ncstore: True
          del_cmipfiles: False

direct_data_root: '/raid/rc40/data/ncs/historical-edr/'
observations_root: '/raid/rc40/data/ncs/obs4comp'
cmip5_root: '/raid/ra40/CMIP5_OTHER_DOWNLOADS/'
```

Save the file and then use the command:

```
validate-configure
```

1.2.4 Taylor Diagram

For this example we want to make a taylor diagram of the atmospheric pressure at the surface for the climatology from 1980 to 2000.



First use the command:

```
validate-configure
```

Then edit the conf.yaml file to the following:

```
run: 'edr'
experiment: 'historical'

defaults:
            dates:
              start_date: '1980-01'
              end_date: '2000-01'
            png: True

plots:
        - variable: 'psl'
          plot_projection: 'taylor'
          data_type: 'climatology'
          comp_obs:
            - 20CR
```

(continues on next page)

(continued from previous page)

```
          comp_models:
            - CanCM4
          comp_ids:
            - cvu

delete:
          del_netcdf: False
          del_mask: True
          del_ncstore: True
          del_cmipfiles: False

data_root: '
observations_root: '/raid/rc40/data/ncs/obs4comp'
cmip5_root: '/raid/ra40/CMIP5_OTHER_DOWNLOADS/'
```

Save the file and then use the command:

```
validate-configure
```

1.3 The validate API

This section describes the **validate** Application Programming Interface (API).

1.3.1 data_loader

THis module contains functions that will load data from netCDF files needed to produce plots. It uses various cdo commands to manipulate the netCDF files if they need to be processed before the data is extracted.

validate.data_loader.**already_calculated**(*name*)

validate.data_loader.**cdos**(*name*, *string*)

validate.data_loader.**dataload**(*ifile*, *var*, *dates*, *realm='atmos'*, *scale=1*, *shift=0*, *remapf='remapdis'*, *remapgrid='r360x180'*, *seasons=None*, *datatype='full'*, *depthneeded=None*, *section=False*, *fieldmean=False*, *gridweights=False*, *cdostring=None*, *yearmean=False*, *external_function=None*, *external_function_args={}*)

Manipulates a file used a series of cdo commands which produce intermediate files, and returns data about the the final file produced based on the specified parameters.

Parameters

ifile [string] the name of the original input file

var [string] variable name

dates [dictionary of the date range as strings of the form 'yyyy-mm'] start_date and end_date keys should be specified

realm [string] realm category (used for masking data) default : 'atmos'

scale [float] scales the data by this value default : 1

shift [float] shifts the data by this valee default : 0

remapf [string] name of the cdo remapping default : remapdis

remapgrid [string] grid to remap the data to default : 'r360x180

seasons [list of strings] seasons to be selected out of ['DJF', 'MAM', 'JJA', 'SON'] None will select all of the seasons default : None

datatype [string] cdo operation to perform all the time axis options are 'climatology', 'trends', 'detrend' anything else not perform any cdo operation default 'full'

depthneeded [list of floats] list of the depths to interpolate the data to in z-axis default : None

section [boolean] set to True to take a zonal mean of the data default : False

fieldmean [boolean] set to True to take a fieldmean of the data default : False

gridweights [boolean] set to True to calculate the area weights of each grid cell

cdostring [string] custom to cdo string to be applied to the input file default : None

yearmean [boolean] take an annual mean of the data before manipulating default : False

external_function [string] name of external function to call default : None

external_function_args [dictionary] keyword arguments to pass to the external function

Returns

numpy array of final data

numpy array of longitudinal cooridinates

numpy array of latitudinal coordinates

numpy array of depths

string of the units

numpy array of the time axis

numpy area of the area weights of the grid cells

validate.data_loader.**depthstring**(*depthlist*)

validate.data_loader.**detrend**(*name*)

validate.data_loader.**field_mean**(*name*)

validate.data_loader.**get_external_function**(*name*)
Returns a function from the external module based on the function name.

validate.data_loader.**get_remap_function**(*remap*)
Returns a cdo function from string of the same name.

validate.data_loader.**grid_weights**(*name*)

validate.data_loader.**intlevel**(*name*, *depthlist*)

validate.data_loader.**mask**(*name*, *realm*)

validate.data_loader.**remap**(*name*, *remapname*, *remapgrid*)

validate.data_loader.**season**(*name*, *seasonlist*)

validate.data_loader.**sel_date**(*name*, *start_date*, *end_date*, *time_average=False*)

validate.data_loader.**sel_var**(*name*, *variable*)

validate.data_loader.**setc**(*name*, *realm='ocean'*)

validate.data_loader.**silent_remove**(*name*)
Removes a file if it exists and does nothing if it doesn't exist

validate.data_loader.**split**(*name*)
Returns the name of a file without the directory path

validate.data_loader.**time_mean**(*name*, *time_average=False*)

validate.data_loader.**trend**(*name*)

validate.data_loader.**year_mean**(*name*)

validate.data_loader.**year_mon_day**(*datestring*)
Seperates a string of from yyyy-mm-dd in to three integers and returns the tuple year,mon,day

validate.data_loader.**zonal_mean**(*name*)

1.3.2 defaults

This module fills the plots with values specified in defaults and fills the remaining options with placeholders so that existence checks will not be needed later.

..moduleauthor:: David Fallis

validate.defaults.**fill**(*plots*, *run*, *experiment*, *cmip6_verification=False*, *defaults={}*)
Fills the blank spaces in plots with default values.

Parameters

plots [list of dictionaries]

model_run [string] run ID

experiment [string] experiment name

defaults [dictionary] values to fill plots

Returns

list of dictionaries

validate.defaults.**filltitle**(*p*)

1.3.3 pdf_organizer

This module contains tools to produce a multipage pdf in an organized and labelled format specific to model plots.

validate.pdf_organizer.**arrange**(*plotnames*)

Outputs a pdf named plots/joined.pdf with all of the plots organized and bookmarked

Parameters

plotnames [list of tuples] (name of plot, plot dictionary, plot type)

validate.pdf_organizer.**orderplots**(*plotnames*)
Organizes the names into a dictionary that can be used to cycle through the plots

Parameters

plotnames [list of tuples] (name of plot, plot dictionary, plot type)

Returns

———-

dictionary

validate.pdf_organizer.**pdfmarks**(*plotdict*)

Writes to pdfmarks file to organize the plots under bookmarks

Parameters

plotdict [dictionary] organized into the bookmark levels

Returns

———-

string with all of the plot names in the order they will be in joined.pdf

1.3.4 taylor

class validate.taylor.**TaylorDiagram**(*refstd*, *fig=None*, *rect=111*, *label='_'*, *srange=(0*, *1.5)*)

Bases: object

Taylor diagram.

Plot model standard deviation and correlation to reference (data) sample in a single-quadrant polar plot, with r=stddev and theta=arccos(correlation).

Methods

add_contours(self[, levels])	Add constant centered RMS difference contours, defined by *levels*.
add_grid(self, *args, **kwargs)	Add a grid.
add_sample(self, stddev, corrcoef, *args, ...)	Add sample (*stddev*,*corrcoeff*) to the Taylor diagram.

add_contours(*self*, *levels=5*, ***kwargs*)

Add constant centered RMS difference contours, defined by *levels*.

add_grid(*self*, **args*, ***kwargs*)

Add a grid.

add_sample(*self*, *stddev*, *corrcoef*, **args*, ***kwargs*)

Add sample (*stddev*,*corrcoeff*) to the Taylor diagram. *args* and *kwargs* are directly propagated to the *Figure.plot* command.

validate.taylor.**test1**()

Display a Taylor diagram in a separate axis.

validate.taylor.**test2**()

Climatology-oriented example (after iteration w/ Michael A. Rawlins).

CHAPTER 2

Introduction

validate is a python package used to produce visulizations and summary statistics of climate model ouput, and its comparison to observations. A set of desired plots and analyses is specified in a simple configuration file. When run validate produces a set of image files, a merged PDF file of all images (fully indexed), as well as a file of summary statistics and a log for reproducibility. Validate is designed to use data in the netCDF file format, specifically as used in the Coupled Model Intercomparison Project (CMIP).

Standard available plots include various maps projections, 2D sections, 1D time-series or line plots, scatter plots and Taylor diagrams. "Comparison" plots compute and display the anomaly between a model run and a specified set of observations. Options such as colorbars, axis limits are assigned by default, but can be easily customized by the user. Currently, validate will compute climatologies and trends over user-specified periods. Validate can be extended to perform any advanced analysis through specification of a user defined external function, which may be written in any language (as long as netCDF is returned).

At the Canadian Centre for Climate Modelling and Analysis (CCCma), validate is used in batch mode to automatically produce a large set of standard summary diagnostic plots at the conclusion of each model run.

2.1 Contributors

David Fallis: davidwfallis@gmail.com

Neil Swart: neil.swart@canada.ca

Pull requests and comments are welcome.

2.2 LICENSE

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

V

Index